Introducing Calabash

automated functional testing for mobile native apps

Karl Krukow,
karl@lesspainful.com,
LessPainful & Trifork
QCon London 2012

About me

About me

BRICS [π-λ] Seminar

- PhD, Computer Science, University of Aarhus, 2006.

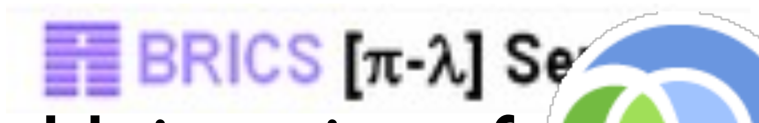
About me

BRICS [π-λ] Seminar

- PhD, Computer Science, University of Aarhus, 2006.
- Developer TRIFORK. for about 6 years mostly Java enterprise, last two years or iOS

About me

- PhD, Computer Science, University of A 2006.
- Developer TRIFORK. for about 6 ye mo ava enterprise, last two years or iOS
- Spare-time Hickey & Clojure fan-boy



About me

- PhD, Computer Science, University of Aarhus, 2006.
- Developer software pilots TRIFORK. for about 6 years mostly Java enterprise, last two years or iOS
- Spare-time Hickey & Clojure fan-boy!
- Co-authoring a book on Dart with Trif
Kresten Krab Thorup.

About me

BRICS [π-λ] Seminar

- PhD, Computer Science, University of Aarhus, 2006.
- Developer software pilots TRIFORK. for about 6 years mostly Java enterprise, last two years or iOS
- Spare-time Hickey & Clojure fan-boy!
- Co-authoring a book on Dart with Trif
  Kresten Krab Thorup.
- Co-owner, iOS responsible at LESSPAINFUL AUTOMATED APP TESTING

Agenda

Agenda

- Automated functional testing for native mobile
    - Some benefits as well as common problems
    - Some desirable properties for a functional testing tool

Agenda

- Automated functional testing for native mobile
  - Some benefits as well as common problems
  - Some desirable properties for a functional testing tool
- Introduce Calabash
  - Focus on iOS

Agenda

- Automated functional testing for native mobile
  - Some benefits as well as common problems
  - Some desirable properties for a functional testing tool
- Introduce Calabash
  - Focus on iOS
- LessPainful: test service and device cloud

Agenda

- Automated functional testing for native mobile
  - Some benefits as well as common problems
  - Some desirable properties for a functional testing tool
- Introduce Calabash
  - Focus on iOS
- LessPainful: test service and device cloud
- Demo!

Functional testing

Functional testing

- Functional and Acceptance tests
  - Actual app, as opposed to an isolated component
  - Often based on use-cases written in natural (domain) language
  - Visual appearance of app screens matter! (Design guidelines, etc)
  - As realistic an environment as practically possible

Functional testing

- Functional and Acceptance tests
  - Actual app, as opposed to an isolated component
  - Often based on use-cases written in natural (domain) language
  - Visual appearance of app screens matter! (Design guidelines, etc)
  - As realistic an environment as practically possible
- For mobile apps, in particular
  - often a manual process: repetitive, expensive
  - Many devices, screens, OS versions, languages

Why automate?

- Save time and effort. Less tedium of repetitive testing with each iteration.
- Higher-quality before app goes to QA and production
  - higher test coverage with fewer resources
  - formalizes test procedure
- Less likely to have regressions.
- Faster feedback for developers.

Some problems...

Some problems...

- *Resources*: Automated test suite is an entire code base that must be developed and maintained?
    - When app changes, test must often change (often, they are deleted instead!).
    - Accuracy - Tests may not be able to express what is wanted? (too precise or too loose)

Some problems...

- *Resources*: Automated test suite is an entire code base that must be developed and maintained?
  - When app changes, test must often change (often, they are deleted instead!).
  - Accuracy - Tests may not be able to express what is wanted? (too precise or too loose)
- *Completeness*: Reduces, but does not eliminate need for manual testing.

Tool Desiderata

(IMO, hope you agree)

Tool Desiderata

(IMO, hope you agree)

- Minimize distance between use cases and actual test code (DSLs?).
- Expressive and efficient to write.
- Extensible
- High-level, declarative (robustness against "minor" UI changes).
- Support testing in realistic environments (multiple real devices, on multiple OS versions, languages).
- Support Continuous integration.

Cucumber

Cucumber and Calabash

Cucumber

- Cucumber is a tool for describing and executing specifications of software
    - specifications are written in a business readable language that is close to natural language.
- Extremely popular tool for test and specs of web applications.
- http://cukes.info/

Cucumber Example

```
Feature:  As an administrator. I want to be able to add and remove users,
          so I can control access to the application

Scenario: Add test user
    When I touch the Add User button
    And I fill in text fields as follows:
      | field        | text    |
      | Last Name    | Knorr   |
      | Username     | knorr   |
    And I touch "Save"
    Then I should be on the Users screen
    And I should see a table containing "Knorr"

Scenario: ...
```

Cucumber Example

```
Feature:  As an administrator. I want to be able to add and remove users,
          so I can control access to the application

Scenario: Add test user
    When I touch the Add User button
    And I fill in text fields as follows:
      | field      | text   |
      | Last Name  | Knorr  |
      | Username   | knorr  |
    And I touch "Save"
    Then I should be on the Users screen
    And I should see a table containing "Knorr"

Scenario: ...
```

Cucumber Example

```
Feature:  As an administrator. I want to be able to add and remove users,
          so I can control access to the application

Scenario: Add test user
    When I touch the Add User button
    And I fill in text fields as follows:
      | field        | text    |
      | Last Name    | Knorr   |
      | Username     | knorr   |
    And I touch "Save"
    Then I should be on the Users screen
    And I should see a table containing "Knorr"

Scenario: ...
```

Cucumber Example

```
Feature:  As an administrator. I want to be able to add and remove users,
          so I can control access to the application

Scenario: Add test user
    When I touch the Add User button
    And I fill in text fields as follows:
      | field       | text    |
      | Last Name   | Knorr   |
      | Username    | knorr   |
    And I touch "Save"
    Then I should be on the Users screen
    And I should see a table containing "Knorr"

Scenario: ...
```

Cucumber Example

```
Feature:  As an administrator. I want to be able to add and remove users,
          so I can control access to the application

Scenario: Add test user
    When I touch the Add User button
    And I fill in text fields as follows:
      | field        | text    |
      | Last Name    | Knorr   |
      | Username     | knorr   |
    And I touch "Save"
    Then I should be on the Users screen
    And I should see a table containing "Knorr"

Scenario: ...
```

Cucumber Example

```
Feature:  As an administrator. I want to be able to add and remove users,
          so I can control access to the application

Scenario: Add test user
    When I touch the Add User button
    And I fill in text fields as follows:
      | field       | text    |
      | Last Name   | Knorr   |
      | Username    | knorr   |
    And I touch "Save"
    Then I should be on the Users screen
    And I should see a table containing "Knorr"

Scenario: ...
```

Step Definitions

- Make the cucumber tests “come alive”
- Written in ordinary programming languages
  - Mostly Ruby (but cucumber-jvm: Java, Clojure,...)

Feature Step definitions

Step Definitions

- Make the cucumber tests “come alive”
- Written in ordinary programming languages
    - Mostly Ruby (but cucumber-jvm: Java, Clojure,...)

Feature

Step definitions

```
Scenario: Add test user
    When I touch the Add User button
...
```

Step Definitions

- Make the cucumber tests "come alive"
- Written in ordinary programming languages
  - Mostly Ruby (but cucumber-jvm: Java, Clojure,...)

Feature

```
Scenario: Add test user
    When I touch the Add User button
...
```

Step definitions

```
When /^I touch the Add User button$/ do
  btn_txt = 'Add user'
  touch("button text:#{btn_txt}")
end
```

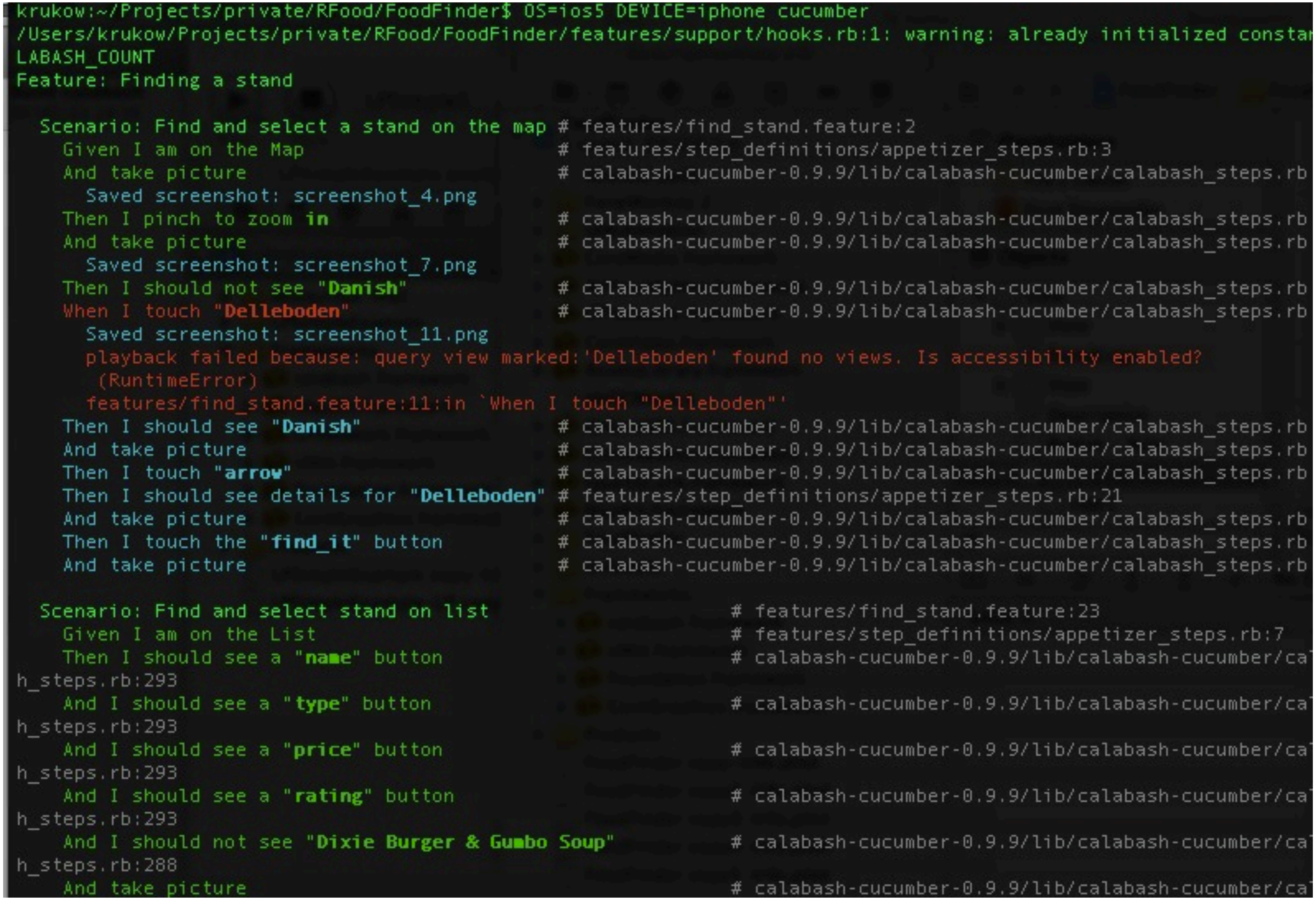
Execution

Execution

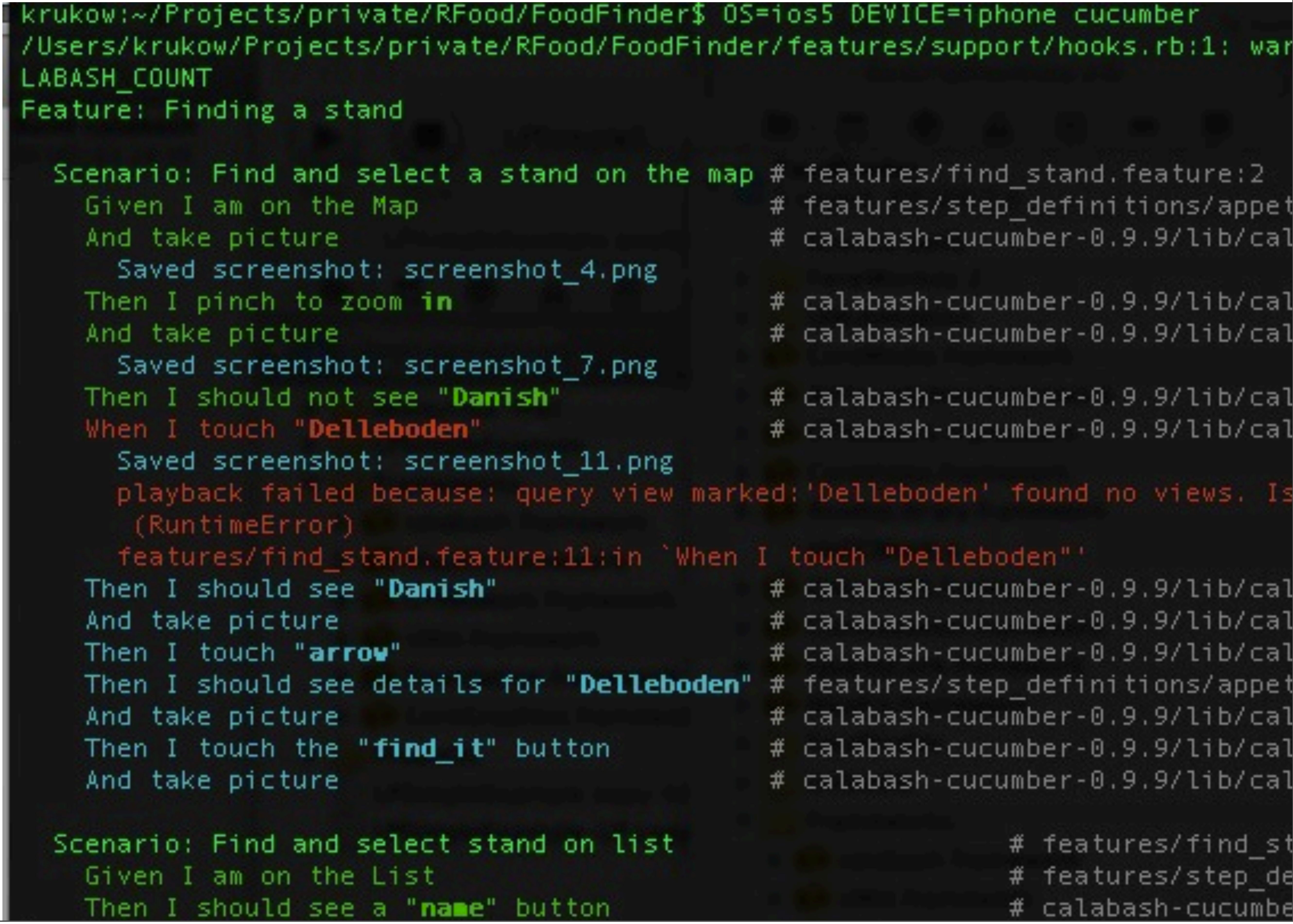
- Executing a test produces a test report
    - for each step, did it succeed or not
    - exception/error message if present

Execution

- Executing a test produces a test report
  - for each step, did it succeed or not
  - exception/error message if present
- Flexible output formats
  - Machine readable (XML, JSON,...)
  - Human readable, console



```
krukow:~/Projects/private/RFood/FoodFinder$ OS=ios5 DEVICE=iphone cucumber
/Users/krukow/Projects/private/RFood/FoodFinder/features/support/hooks.rb:1: warning: already initialized constan
LABASH_COUNT
Feature: Finding a stand

  Scenario: Find and select a stand on the map # features/find_stand.feature:2
    Given I am on the Map                      # features/step_definitions/appetizer_steps.rb:3
    And take picture                           # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
      Saved screenshot: screenshot_4.png
    Then I pinch to zoom in                    # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
    And take picture                           # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
      Saved screenshot: screenshot_7.png
    Then I should not see "Danish"             # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
    When I touch "Delleboden"                  # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
      Saved screenshot: screenshot_11.png
      playback failed because: query view marked:'Delleboden' found no views. Is accessibility enabled?
       (RuntimeError)
      features/find_stand.feature:11:in `When I touch "Delleboden"'
    Then I should see "Danish"                 # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
    And take picture                           # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
    Then I touch "arrow"                       # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
    Then I should see details for "Delleboden" # features/step_definitions/appetizer_steps.rb:21
    And take picture                           # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
    Then I touch the "find_it" button          # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb
    And take picture                           # calabash-cucumber-0.9.9/lib/calabash-cucumber/calabash_steps.rb

  Scenario: Find and select stand on list                        # features/find_stand.feature:23
    Given I am on the List                                       # features/step_definitions/appetizer_steps.rb:7
    Then I should see a "name" button                            # calabash-cucumber-0.9.9/lib/calabash-cucumber/ca
h_steps.rb:293
    And I should see a "type" button                             # calabash-cucumber-0.9.9/lib/calabash-cucumber/ca
h_steps.rb:293
    And I should see a "price" button                            # calabash-cucumber-0.9.9/lib/calabash-cucumber/ca
h_steps.rb:293
    And I should see a "rating" button                           # calabash-cucumber-0.9.9/lib/calabash-cucumber/ca
h_steps.rb:293
    And I should not see "Dixie Burger & Gumbo Soup"             # calabash-cucumber-0.9.9/lib/calabash-cucumber/ca
h_steps.rb:288
    And take picture                                             # calabash-cucumber-0.9.9/lib/calabash-cucumber/ca
```

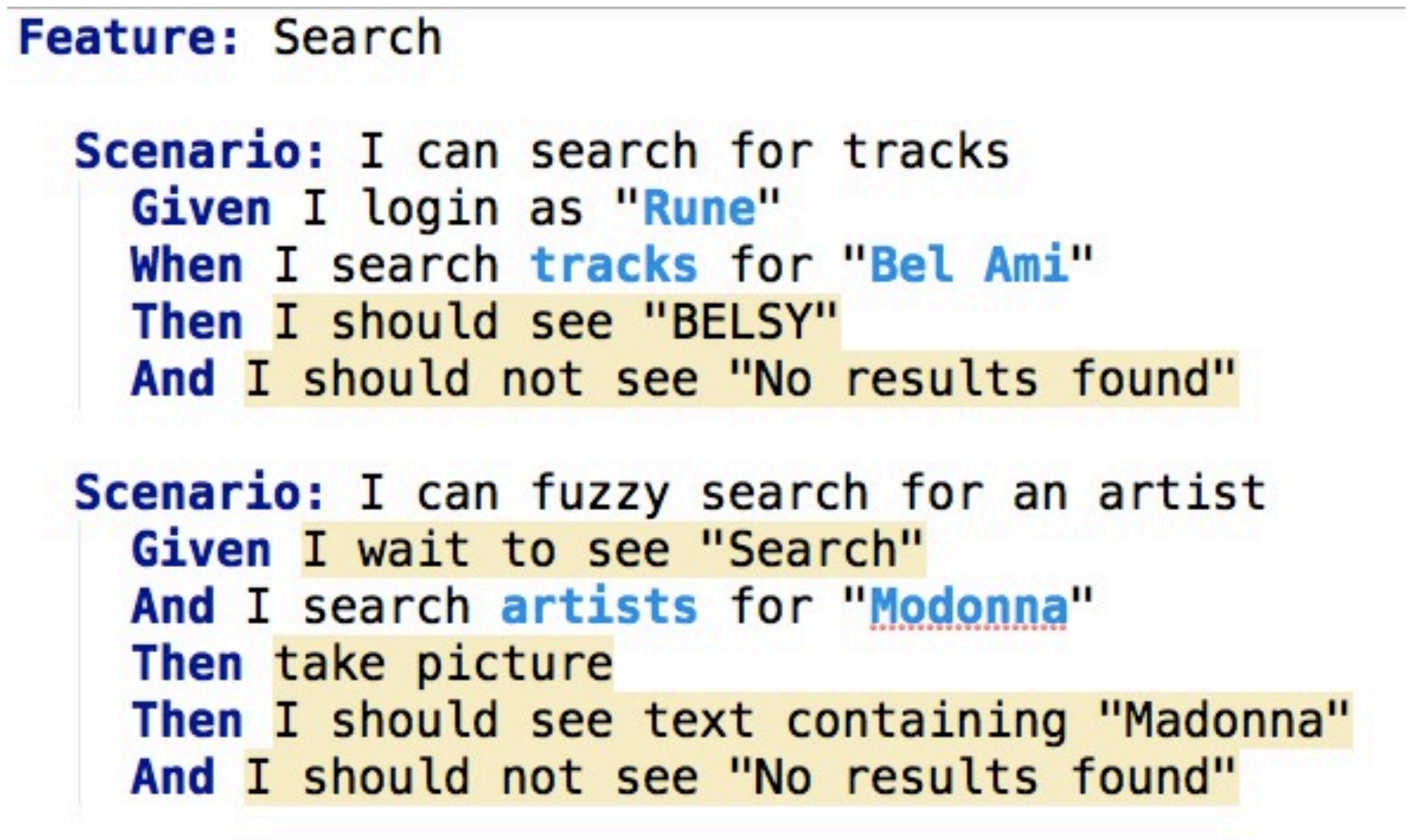


```
krukow:~/Projects/private/RFood/FoodFinder$ OS=ios5 DEVICE=iphone cucumber
/Users/krukow/Projects/private/RFood/FoodFinder/features/support/hooks.rb:1: war
LABASH_COUNT
Feature: Finding a stand

  Scenario: Find and select a stand on the map # features/find_stand.feature:2
    Given I am on the Map                      # features/step_definitions/appet
    And take picture                           # calabash-cucumber-0.9.9/lib/cal
      Saved screenshot: screenshot_4.png
    Then I pinch to zoom in                    # calabash-cucumber-0.9.9/lib/cal
    And take picture                           # calabash-cucumber-0.9.9/lib/cal
      Saved screenshot: screenshot_7.png
    Then I should not see "Danish"             # calabash-cucumber-0.9.9/lib/cal
    When I touch "Delleboden"                  # calabash-cucumber-0.9.9/lib/cal
      Saved screenshot: screenshot_11.png
      playback failed because: query view marked:'Delleboden' found no views. Is
       (RuntimeError)
      features/find_stand.feature:11:in `When I touch "Delleboden"'
    Then I should see "Danish"                 # calabash-cucumber-0.9.9/lib/cal
    And take picture                           # calabash-cucumber-0.9.9/lib/cal
    Then I touch "arrow"                       # calabash-cucumber-0.9.9/lib/cal
    Then I should see details for "Delleboden" # features/step_definitions/appet
    And take picture                           # calabash-cucumber-0.9.9/lib/cal
    Then I touch the "find_it" button          # calabash-cucumber-0.9.9/lib/cal
    And take picture                           # calabash-cucumber-0.9.9/lib/cal

  Scenario: Find and select stand on list                # features/find_st
    Given I am on the List                               # features/step_de
    Then I should see a "name" button                    # calabash-cucumbe
```

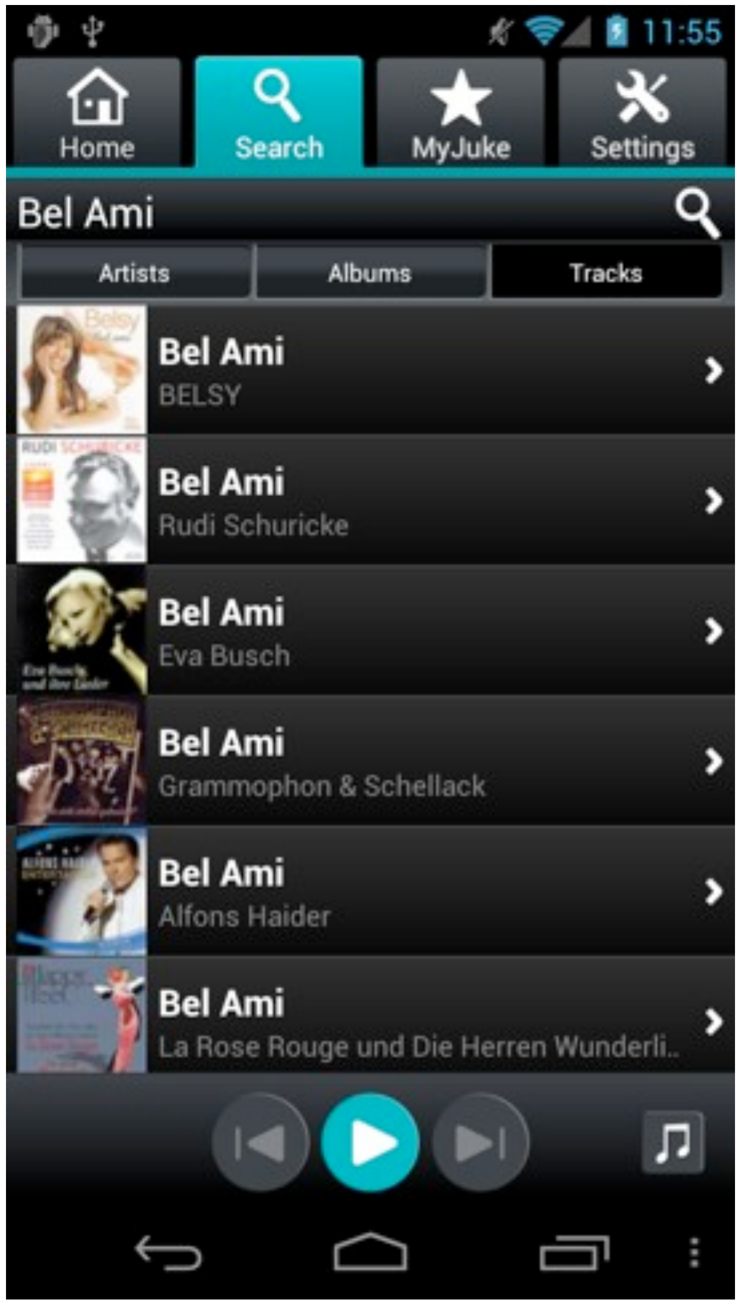
Calabash

- New open source project automated functional testing of Android and iOS apps.
- *One* interface: Cucumber, for Android and iOS tests.
  - Predefined and custom steps (Ruby + soon: JVM).
  - Reuse of tests across platform possible.
- Run on physical devices and simulators.
- Support for hybrids ("embedded webviews") (WIP)
- Options: Run in a device cloud using the LessPainful service, commercial support.

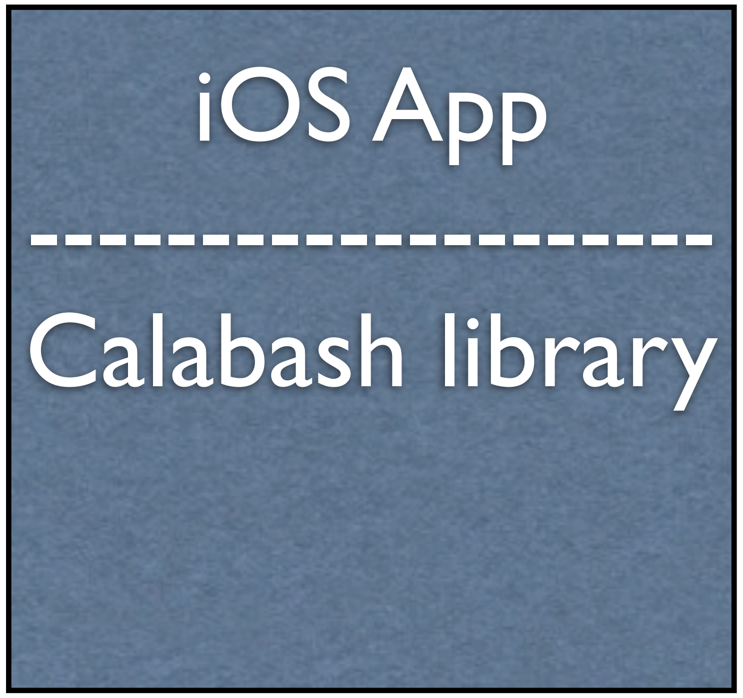
Example: 24/7e, Juke



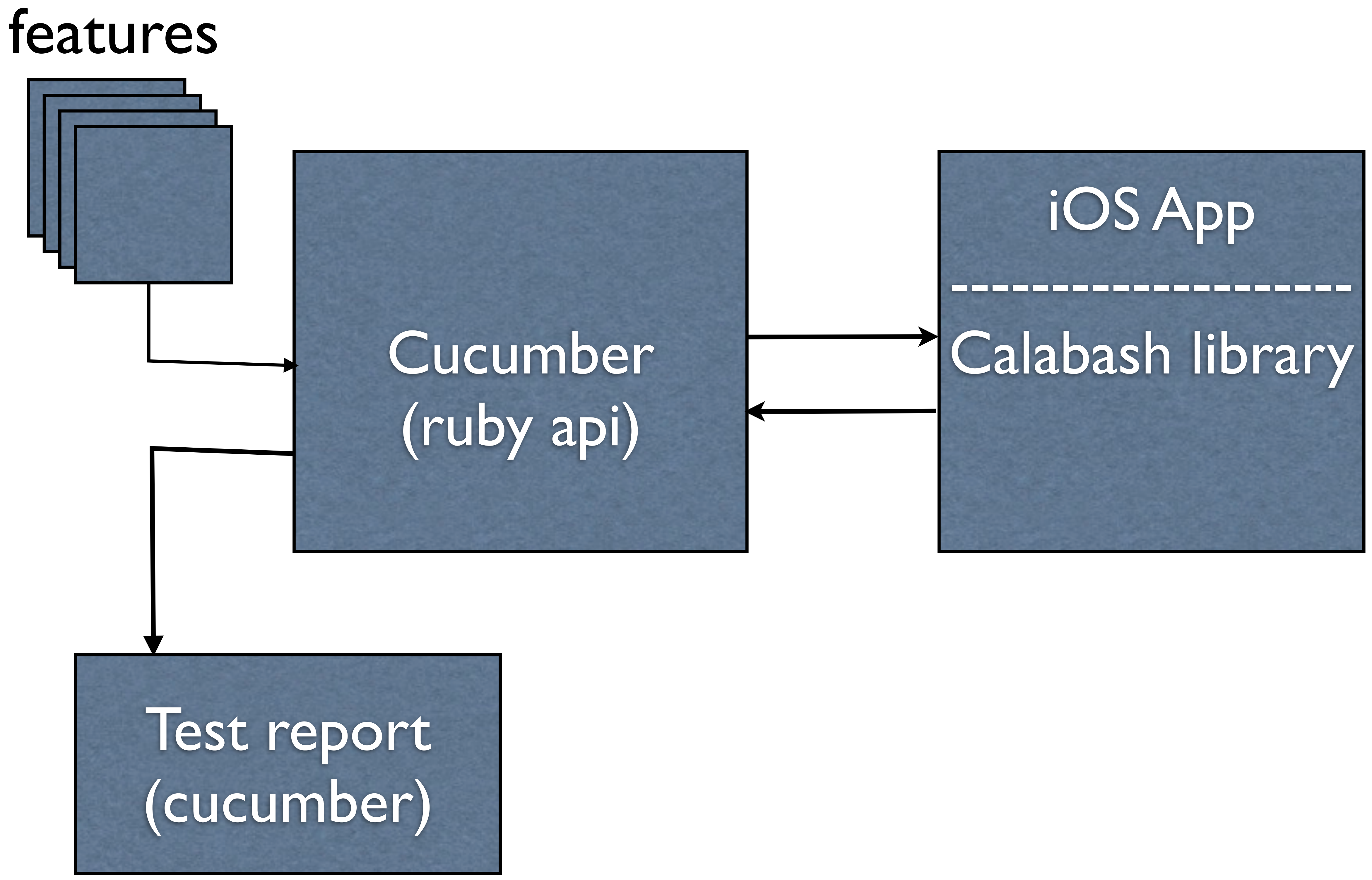
```
Feature: Search

  Scenario: I can search for tracks
    Given I login as "Rune"
    When I search tracks for "Bel Ami"
    Then I should see "BELSY"
    And I should not see "No results found"

  Scenario: I can fuzzy search for an artist
    Given I wait to see "Search"
    And I search artists for "Modonna"
    Then take picture
    Then I should see text containing "Madonna"
    And I should not see "No results found"
```



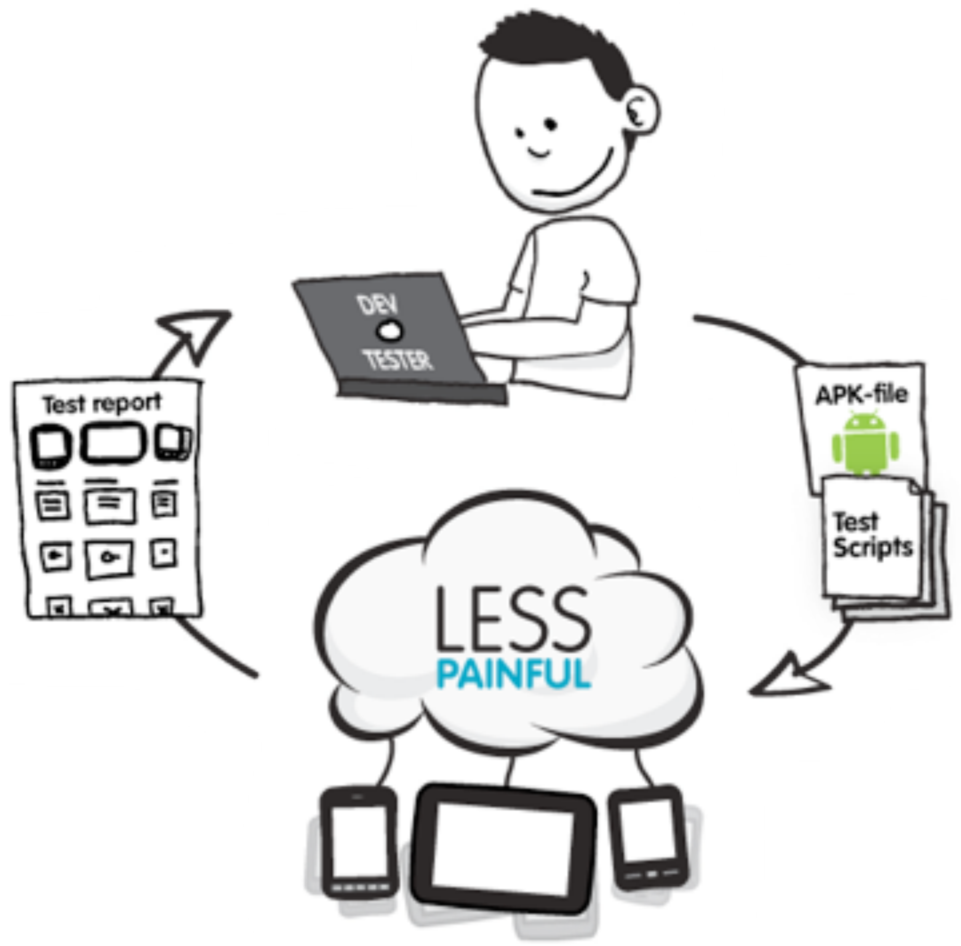
Architecture iOS



Architecture iOS



LessPainful
Test Execution in the Cloud



- Execute Calabash tests concurrently on many devices, OS'es, languages.
- Authentic: Not jailbroken, iOS and Android devices, rotation.
- Visual test reports.
  - Comparison across models and operating systems.
- Continuous integration.

Mobile Test Lab

www.lesspainful.com

A bit about the LessPainful architecture

A bit about the LessPainful architecture

- Distributed system components:
    - Website (app+test submission, test reports)
    - AWS S3 storage (never deleted)
    - Queing (Resque)
    - Physical devices and device hosts

A bit about the LessPainful architecture

- Reliability and availability: Asynchronous, queue based
  - Queue for each device model and OS
  - Several equivalent physical devices can take off same queue
  - Tests not lost if down or device not ready
  - Failures can be retried
  - Cannot depend on timings

Calabash iOS: more detail

Calabash iOS: more detail

- Very easy to get started for iOS developers.
- Declarative query language for finding views.
  - Based on UISpec, but *simplified,* extended.
- Touch synthesis, supports multitouch gestures. Extensible.
- Interactive, exploratory development experience.
- Uses device accessibility for identifying views.

Query

Query

- Declarative language for finding UI components.
  - Syntax and semantics based on UISpec (GPLv3).
  - New implementation (EPL license).

Query

- Declarative language for finding UI components.
    - Syntax and semantics based on UISpec (GPLv3).
    - New implementation (EPL license).
- Queries are like CSS selectors or XPath
    - `label`, `label text:'Hello'`, `label index:2`
    - `view marked:'thepane' label`
    - `label {text LIKE 'Hel*'}`
    - `label text:'foo' parent tableViewCell`

Touch synthesis

- Gestures can be represented as lists of dictionaries. (eventtype, x,y,...)
- Uses private iOS API to perform such sequences of touch events.
- Goes through all the phases that are activated when users performs touchs (i.e., no synthetic calls to gesture rec. etc).
- Combined with queries, can be used to synthesize user actions on views.

Prototype gestures

- Calabash iOS has a number of built-in event sequences that can be played back.
  - (touch, swipe, pinch, etc)
  - Events can be relocated (translated) to different views, and optionally offsets.
- Extensible: you can record your own gestures if none of the built in suits you.

Demo:
- Calabash iOS
- LessPainful Device Cloud

iOS Comparisons

- Several options available. To my knowledge:
  - Calabash
  - UIAutomation, Apple
  - Zucchini, iOS Testing Framework
  - Frank, Pete Hodgson, ThoughtWorks
  - UISpec, http://code.google.com/p/uispec/
  - FoneMonkey => MonkeyTalk, GorillaLogic
  - KIF, Square
  - NativeDriver, http://code.google.com/p/nativedriver/

References

- https://github.com/calabash
  - https://github.com/calabash/calabash-ios
  - https://github.com/calabash/calabash-ios/wiki
  - https://github.com/calabash/calabash-ios-server
- http://blog.lesspainful.com/
- https://www.lesspainful.com/

Summary

- Calabash iOS tradeoffs and mitigations
  - DSLs: Cucumber, and Query language (decl, high level)
  - Full power of Ruby in tests (soon JVM langs too)
  - Advanced touch synthesis
    - recordings give some brittleness, which is
    - mitigated by supporting relocating and manipulating of recorded event sequences (which are just lists of dictionaries).
  - Good interactive developent experience
  - LessPainful provides a device cloud and test execution as service
  - Requires linking of a framework
  - Not good for some games (randomness, “gameplay” & “feel”, timing critical)
  - Not good for Phone-to-Phone coordination (call, text, bluetooth)

Making app testing less painful...

http://www.lesspainful.com